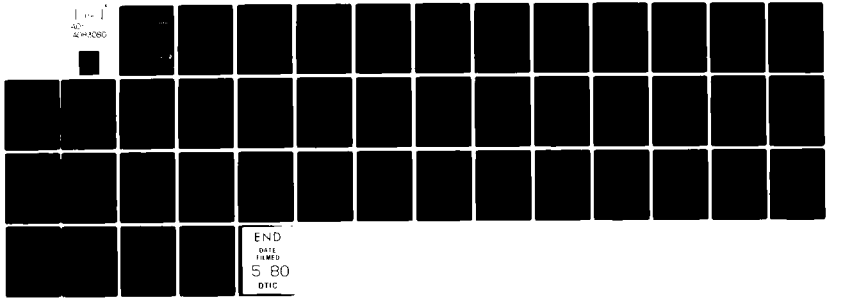
AD-A083 080 BOLT BERANEK AND NEWMAN INC CAMBRIDGE MA F/G 9/4
EFFICIENT ENCODING AND DECODING OF SPEECH.(U)
APR 80 M KRASNER, M BEROUTI MDA904-79-C-0411
UNCLASSIFIED BBN-4355 NL



Bolt Beranek and Newman Inc.

BBN Report No. 4355
Contract No. MDA904-79-C-0411

EFFICIENT ENCODING AND DECODING OF SPEECH

Quarterly Progress Report No. 1
1 November 1979 - 31 January 1980

Submitted to:

Mr. David Fonseca, R814
9800 Savage Road
Fort George G. Meade, MD 20755

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER BBN Report No. 4355	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Efficient Encoding and Decoding of Speech		5. TYPE OF REPORT & PERIOD COVERED 1st Quart. Progress Report Nov. 1979 - Jan. 1980
		6. PERFORMING ORG. REPORT NUMBER Report No. 4355
7. AUTHOR(s) M. Krasner M. Berouti		8. CONTRACT OR GRANT NUMBER(s) MDA904-79-C-0411
9. PERFORMING ORGANIZATION NAME AND ADDRESS Bolt Beranek and Newman Inc. 50 Moulton St., Cambridge, MA 02138		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 12 441
11. CONTROLLING OFFICE NAME AND ADDRESS Maryland Procurement Office 9800 Savage Road Fort George A. Meade, MD 20755		12. REPORT DATE April 1980 13. NUMBER OF PAGES 40
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Commerce for sale to the general public.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Speech compression, linear prediction, adaptive predictive coding, down-sampling, quantization, pitch filter, preemphasis, noise shaping, variable-rate output.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

In this first quarterly progress report we discuss our work on converting the output of the adaptive predictive coder into a fixed-rate bit-stream, and on improving the coded speech quality by means of various noise shaping filters. We also discuss the computational complexity of the algorithm.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Report No. 4355

EFFICIENT ENCODING AND DECODING OF SPEECH

Quarterly Progress Report No. 1
1 November 1979 to 31 January 1980

Prepared by:

Bolt Beranek and Newman Inc.
50 Moulton Street
Cambridge, Massachusetts 02138

Prepared for:

Mr. David Fonseca, R814
9800 Savage Road
Fort George G. Meade, MD 20755

TABLE OF CONTENTS

1. INTRODUCTION

During the first quarter of this contract (on the efficient encoding and decoding of speech under noisy channel conditions) we began work on various aspects of the system. The two basic concepts underlying our work thus far are: (i) Under channel errors, the receiver looses frame synchronization and becomes unable to synthesize speech, and (ii) it may be beneficial to provide the system parameters with some protection against channel errors.

The problem of loss of frame synchronization is due to the use of variable-length codes. We reexamined the benefits of using entropy coding and our work on that subject is reported in Section 1. Having ascertained that the use of a fixed self-synchronizing code is beneficial to the system, we then proceeded to implement a solution to the problem of loss of frame synchronization. Our solution is discussed in Section 2.

As for protecting the system parameters against channel errors, our previous experience on the subject, from work done under other government contracts, indicates that about 10% of the channel capacity must be freed to be used for parameter protection. The 10% decrease in bit rate is provided by decreasing the number of bits available to encode the residual waveform. We anticipate that such action will cause a

degradation in the quality of coded speech. However, there are several possible methods to enhance the quality of the coded speech, in an effort to counteract the decreased available bandwidth for the coding of the residual. Our work on such methods is still on-going. In this report we discuss two methods for which we have already obtained some results. The first method is time domain noise shaping and is discussed in Section 4 and the second method is frequency domain noise shaping and is discussed in Section 5. In our experiments we have used a data base of clean speech utterances. Although we are aware that the present APC system will be used to encode noise-corrupted speech, it was necessary to perform the initial experiments with noise-free speech. The reason is that it is difficult to assess the effect of a particular method on the quality of coded speech in the presence of a noise background. We plan to use the noise-corrupted speech data base in our future work.

In this report we also discuss the computational complexity of the algorithm in Section 6, and describe our plans for the future in Section 7.

2. INVESTIGATION OF THE SELF-SYNCHRONIZING CODE

2.1 Comparison of Self-Synchronizing and Optimal Codes

Use of self-synchronizing codes have the primary advantage that the bit stream is uniquely decodable starting in the middle of a codeword sequence. When used on a digital channel that causes transmissions errors, the self-synchronizing property implies that an error will not effect the decoding of code words subsequent to that error. In general, however, the optimal coding will not be self-synchronizing. Thus, there is a tradeoff involved in the use of a self-synchronizing code. In particular for this APC system, we investigate whether the cost of the additional bit rate for the coding by a self-synchronizing code is offset by the advantages of the synchronization properties. In this section, the bit rate required by the self-synchronizing code described in the proposal [1] is compared to the bit rate required by other one-dimensional coding schemes under the constraint of a given quantization distortion. Specifically, that self-synchronizing code is compared to a Huffman code, the optimal code with the restriction of binary coding single source words, and to the entropy, the lower limit to what may be achieved by any code of arbitrary block length (i.e. allowing the coding of discrete source words in blocks of length greater than one.) Note that although a block source code may not be

appropriate when transmitting over a channel with errors, it represents the limit to possible performance by coding schemes and, therefore, is an important scheme for comparison.

In addition to the above advantages, the self-synchronizing code is fixed before coding and does not vary as a function of the input. Schemes that are more efficient such as Huffman codes, will require the transmission of auxiliary information to specify the distribution of the quantized values necessary for generation of the coding tables. The purpose of this comparison is to see if the additional bits required in coding by a self-synchronizing as compared to an optimal code are justified by the savings due to the lower overhead inherent in a use of fixed set of codes and by its performance advantages.

For this comparison, it is convenient to separate the coding problem into two parts: quantization of single samples into a finite set of source words under the criterion of a given distortion, and use of a source coding scheme to assign a sequence of binary code words to the sequence of source words. Under the assumption that the number of levels used in the quantization is large and that the error is small, the optimal quantizer has threshold levels that are uniformly spaced [2]. This assumption does not hold for quantization with average rates of approximately two bits per sample as is necessary for this

encoding system. It has been shown, however, that the uniform quantizer will still be optimal for exponential and Laplacian distributions, and will be very close to optimal for other distributions [3]. For this reason, we consider a uniform quantizer and allow only the spacing between threshold levels to be varied according to the input signal level.

Each of eight utterances were processed using several different coding schemes for a given quantization distortion. The results are shown in Table 1. Use of a Huffman code specified once per utterance yields a savings of only 0.003 bits per sample relative to the self-synchronizing code. Implementing a block code to achieve a rate closer to the entropy could save a maximum of 0.038 bits per sample. As the distribution of the quantization sample values is not stationary on a frame-to-frame basis, more efficient coding can be achieved by using the distribution in each frame. When specification of the quantization sample distribution is permitted for each frame, a set of Huffman codes can now produce a 0.059 bits per sample savings. Block source coding in each frame could yield a maximal savings of 0.120 bits per sample. This would correspond to an approximate improvement of 0.72 dB in the signal to noise ratio. Note that this bound on coder performance is not achievable since implementation of such a scheme would require transmission of certain necessary overhead information that was not included in

Code Type	Bits/Sample	Savings in Bits/Sample over Self-sync Code	Code Rate for Residual per second	Savings in Bits/Second over Self-sync Code
Self-synchronizing	2.147	-	14313	-
Huffman based on utterance	2.144	0.003	14292	20
Entropy of utterance	2.109	0.038	14060	253
Average of Huffman based on each frame	2.088	0.059	13920	393
Average entropy of frames	2.027	0.120	13513	800

TABLE 1. COMPARISON OF VARIABLE LENGTH CODES

the above bound calculation. As the maximal improvement possible by use of another coding scheme is less than 1 dB, we conclude that the self-synchronizing code is nearly optimal and has several advantages over more complex source coding schemes such

as increased invulnerability under channel errors, a fixed set of codes that does not require the transmission of overhead information.

2.2 Variation of Bit Rate by Frame

Using variable length codes, the number of bits used to encode a frame is a function of the quantization sample value distribution. For most systems, however, it is necessary to transmit over a channel with a fixed rate. A scheme that uses a channel buffer to convert for such a channel often incorporates a delay that may not be tolerable. Another method, discussed in the following section, is to force the number of bits used in the encoding to a fixed amount in every frame. Frames of the speech signal that were coded with fewer than the average number of bits by the variable length coding scheme will receive more bits for coding in this method and, hence, will have less encoding degradations. Conversely, frames that had used more bits than average will be forced to used fewer bits and will have greater encoding errors. The intent of this section is to investigate this variation of the number of bits per sample used by the variable length code in each frame. The next section will discuss modification of the coding algorithm to allow for use on a fixed capacity channel.

Two factors affect the number of bits that are used to encode a frame. The first is caused by the method of choosing the spacing of the quantizer threshold levels for each frame, or equivalently, the method of determining the gains used before and after a fixed uniform quantizer. This gain is calculated from the energy in the residual after the open loop, all-zero analysis filter. In the APC loop, the signal at the quantizer is comprised of that residual plus a term of the filtered quantization error. If this error term is within 10 dB of the residual energy, the energy at the input to the quantizer will increase by at least 1 dB. This change in the variance of the distribution for which the quantizer was designed will cause the number of bits used in a frame to increase. The second factor is the actual shape of the distribution of values in a frame. Different distributions may cause either an increase or a decrease in the number of coding bits.

Experimental results show both factors to be important. We have observed that some frames use up to 25% more bits than average. These frames tend to be associated with large prediction gains as measured by Vp in the linear prediction analysis. If the variance at the input to the quantizer is forced to unity by iteration, the number of extra bits used in these frames will be decreased by approximately half. This indicates that for those frames, it is both the variance and

shape of the distribution that are important. It is rare, however, for the variance of the actual distribution to be less than the variance of the residual. For such frames, it is primarily the shape of the distribution that would cause those frames to use less bits than average. Almost without exception, those frames that have been observed to use significantly less bits are frames during transition from silence before a stop consonant to the consonant burst. The abrupt change of energy within the frame generates a distribution that uses up to 40% less bits than the average.

For ease of transmission, it is desired to have a fixed number of bits used in the coding of each frame. By forcing each frame to use the same number of bits for coding, frames that use more bits than average will lose bits and have higher noise levels. Conversely, frames that use less than average will be allowed more bits and have lower noise levels. A method for forcing the number of bits used in each frame to a constant is discussed in the next section.

3. FRAME SYNCHRONIZATION AND THE FIXED RATE CHANNEL

In the proposal [1] for this work, we have discussed at length the effects of channel errors on the performance of the system. In particular, we have distinguished between two major aspects of the problem: sample synchronization and frame synchronization. At present, we believe that, due to the use of self-synchronizing codes, sample synchronization is not a major problem. In this section, we explain one possible solution we have explored to solve the problem of loss of frame synchronization.

As mentioned in Section 2.2, the basic idea in our method is to force the number of bits used in each frame to be a constant, while still using variable-length codes. For example, each frame is encoded into 39 bits for parameters and 361 bits for the residual samples. In the proposal, we presented a similar idea, except that we allowed the size of the block to be encoded into a fixed number of bits to be a certain integer number of frames. During this quarter, we investigated the simplest approach, that of having one frame per block. At a frame rate of 40 frames/s, the above 400 bits correspond to a total fixed rate of 16,000 b/s. In this method, the output bit rate for the system is fixed or constant in time. Although a fixed output rate is desirable for transmission over fixed-capacity synchronous channels, the

principal aim of the method here is to avoid the loss of frame synchronization. To understand this aspect of the method, recall that, under channel errors, the number of decoded samples at the receiver may vary in each frame, i.e., is not guaranteed to be the same as the number of transmitted samples. Thus, in a free running variable rate system under channel errors, the receiver can count neither bits nor samples to determine when the decoding of residual samples must terminate and the decoding of the parameters of the next frame must start. However, when the number of bits used each frame to encode the residual samples is fixed, the decoder can count the number of bits received, e.g. 400, and can start decoding the parameters of the next frame without ever loosing synchronization.

The method to force the number of bits to be a constant at each frame is to change the gain factor in front of the quantizer and repeat the analysis of the APC loop for the whole frame. The new residual thus obtained is encoded and the number of bits used is counted again. The process is repeated iteratively until convergence is achieved. Such a method relies heavily on the 6 dB per bit rule discussed in the final report [4] of the previous contract with our sponsor. The 6 dB per bit rule is a relationship between the change in entropy (or bit rate) in bits per sample and the change in the gain factor (or variance of the input to the quantizer) in decibels. Intuitively, we expect

that, with a fixed step size quantizer, the entropy or bit rate can be changed by changing the variance of the input to the quantizer, which, in turn, can be adjusted by changing the gain factor.

In our recent experiments we have modified the iterative algorithm such that we depend on the 6 dB per bit rule only during the initial steps of the method, until a "zero-crossing" is reached. We consider that a zero-crossing has been located when the total number of bits used in a frame changes, from one iteration to the next, from being more than, to being less than the desired fixed number of bits. At this point the algorithm becomes the "modified false position" method for locating a zero of a function [5]. The method is illustrated in Fig. 1.

In the figure, the solid curve is the hypothetical relationship between the number of bits used in a frame and g, the gain factor in decibels. The ordinate B(g) is the difference between the actual number of bits used and the target value. In general, the target value is taken to be the desired fixed number of bits, $\bar{B}$, e.g., 361 in the previously mentioned example. The figure illustrates the steps of the algorithm as follows. The abscissa g_1 is the initial estimate of the gain; its use in the APC loop yields a positive value $B(g_1)$, indicating that the number of bits used is greater than desired. At the second

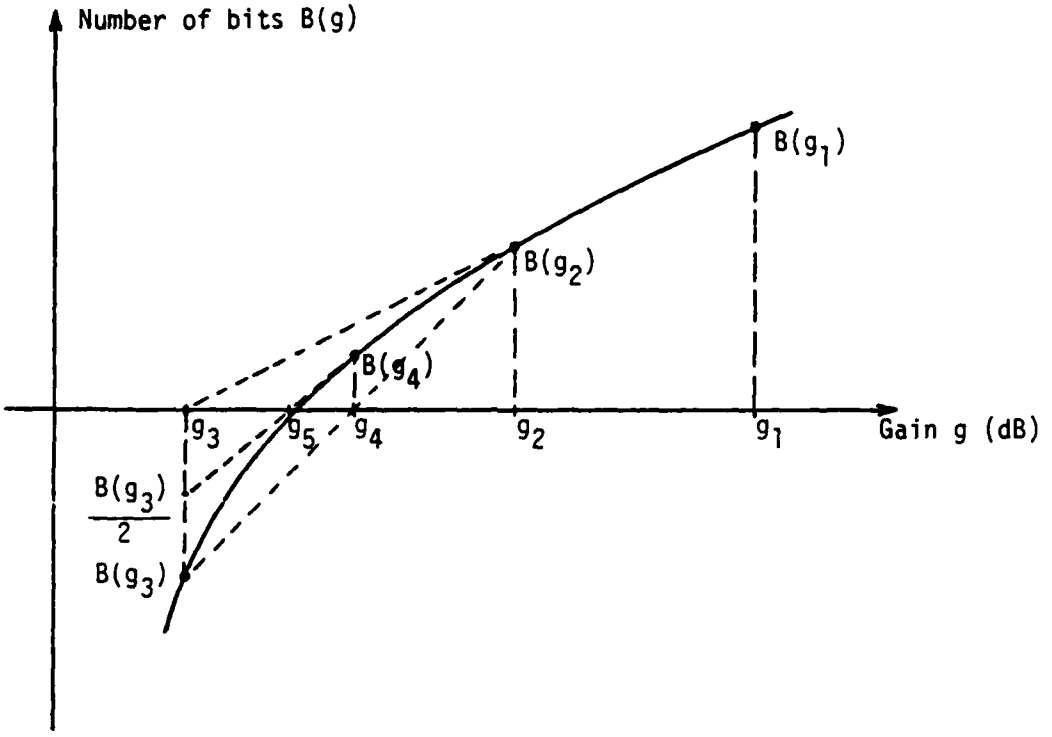


FIG. 1. NUMBER OF BITS USED PER FRAME VERSUS GAIN VALUE, ILLUSTRATING THE ITERATIVE CONVERGENCE ALGORITHM TO CONVERT TO A FIXED NUMBER OF BITS PER FRAME.

iteration, a new value of gain, g_2, is derived from the 6 dB per bit rule. It is obtained from

$$\Delta g = g_2 - g_1 = -6\ B(g_1)/N \qquad (1)$$

where N is the number of samples in a frame. With this new value for gain, the APC residual is computed and encoded again. The figure illustrates the case where, at the second iteration, the number of bits still exceeds the target value ($B(g_2)$ positive). A third iteration is needed, where the gain value g_3 is again given by (1), with the indices 1 and 2 replaced by 2 and 3. At the third iteration, $B(g_3)$ is negative. At this point a

zero-crossing has been located and the algorithm changes from the 6 dB per bit rule to the false position method. In this second method, an estimate of the location of the zero of the function B(g) is obtained from

$$g_4 = [g_2B(g_3) - g_3B(g_2)]/[B(g_3) - B(g_2)] \qquad (2)$$

The new gain value g_4 is now used in the APC loop. Fig. 1 illustrates the case where $B(g_4)$ is positive. Thus the zero-crossing must be between the abscissas g_4 and g_3, and a new value, g_5, is now derived from (2), with the appropriate indices. In general the last two known points to be used in (2) must always be such that a zero-crossing occurs between them. For example, had $B(g_4)$ been negative, the zero-crossing would have been between g_4 and g_2 which must be used in (2) instead of g_4 and g_3.

In practice, the algorithm we implemented differs from the one described above in three ways. First, after the points $[g_1, B(g_1)]$ and $[g_2, B(g_2)]$ have been located, the third iteration gain, g_3, is derived by passing a line through those two points. The abscissa g_3 is at the intersection of the line with the horizontal axis. We feel that passing a line through two known points is preferable to passing a line through $[g_2, B(g_2)]$ with a fixed slope of 6 dB per bit. The straight line approximation with two known points is used at each subsequent iteration until a zero crossing is reached.

The second change to the algorithm is designed to improve its convergence characteristics. Recall from the previous paragraph that the point $[g_3, B(g_3)]$ is used in two successive iterations. It is used in the fourth iteration to derive point g_4, between g_2 and g_3, and in the fifth iteration to derive g_5, between g_4 and g_3. A scheme to improve the convergence properties of the false position method is to divide by two the ordinate of any point being used twice. Thus to derive g_5, we use $B(g_3)/2$ and $B(g_4)$. Should the point $[g_3, B(g_3)/2]$ be needed in the next iteration, its ordinate will be divided by two, e.g., $B(g_3)/4$ and $B(g_5)$ will be used to derive g_6.

The third change in the algorithm is to set the target number of bits to be less than the *desired fixed* total $\bar{B}$. In practice, we have used a target value of about 5 bits less than $\bar{B}$. The reason for this change is that, at the end of the maximum allowed number of iterations, we choose the value g_i, which yields a total number of bits closest to and less than $\bar{B}$. Such a value could in fact be larger than the target used in the iterations, but is less than $\bar{B}$. It is rare that one of the iterations yields exactly $\bar{B}$. Therefore, after picking the best choice, one must transmit a few additional bits to make up the difference. We call these: filler bits.

Finally, we have observed that very small changes in the

gain factor, of the order of 0.1 dB, can lead to large changes in the number of bits used in a frame. For that reason, we decided to use the unquantized value of the gain in front of the quantizer, while still retaining the quantized value in the feedback path of the APC loop. The gain, however, is now being encoded into 9 bits, instead of 6, to minimize the effect of the mismatch between the quantized and the unquantized values of gain on the performance of the APC loop.

Our experiments done with 10 clean-speech sentences show a drop of about 1 dB in the output signal-to-noise ratio (SNR) relative to the free running case where the conversion to fixed rate is not done. However, our informal listening tests show that one can distinguish between the two coded versions only with great difficulty. Furthermore, it is difficult to determine which of the processed utterances is preferable and which one sounds closer to the original input speech. We attribute this difficulty to the fact that, as pointed out in Section 2.2, the conversion to fixed rate improves some frames while degrading the quality of others.

For a maximum of 5 iterations allowed at each frame we have observed that the system uses an average of 7 filler bits per frame, a negligible amount. We believe that the 1 dB drop in SNR is mostly due to the gain mismatch reported above. Thus, we

conclude that the proposed solution to frame synchronization is an efficient one, although it is computationally expensive. Further discussion about the computational complexity of the complete APC system is given in Section 6.

The final experimental observation to be reported in this section is that, under certain conditions, the algorithm fails to converge at some frames. We were able to trace the problem back to the fact that it was not possible to obtain a zero crossing, even with an unlimited number of iterations. Some frames start by being encoded into a number of bits larger than the available $\bar{B}$. Under certain conditions, it is not possible to find a gain value g that yields fewer bits than $\bar{B}$, with the exception of the case g=0 for which all samples are quantized into zero and transmitted with 1 bit each. The reason for this problem is not due to a weakness in the above described convergence algorithm but, instead, is directly related to the stability properties of the APC loop. Since the APC loop is a feedback system, excessive noise feedback may jeopardize its stability. The problem of instabilities in the APC loop is discussed in Section 5.4.

4. TIME DOMAIN NOISE SHAPING

In any APC system, the encoding error is non-stationary, varying as a function of the input signal. By varying the functional blocks of the system, two basic attributes of the error can be controlled, the variations of noise power in time and the noise spectral shape. It is very difficult, a priori, to determine the optimal variations of those objective parameters from the viewpoint of subjective quality. This section describes experimental results in controlling the noise power in time, time domain noise shaping.

Time domain noise shaping can be categorized by the rate at which the noise power is controlled. We have experimented with shaping of the noise within each frame by an adaptive bit allocation scheme that does not change the number of bits used in that frame. Shaping on a longer time basis was not attempted as it is not compatible with the variable to fixed rate algorithm described previously.

The changes in noise power and spectrum are controlled by the input signal energy and signal short-time spectrum. In an APC system without spectral noise shaping, i.e., where the encoding error is a white noise process, the noise power is determined by the signal energy and the prediction gain during the analysis time period. For an APC system with spectral noise

shaping, both noise power and spectral density will be functions of the input signal energy and short-time spectrum during the analysis time period. The rate of adaptation of the noise is fixed by the rate at which the system parameters corresponding to spectral and energy information are updated. Because of data rate considerations, we have set the update rate of the linear prediction filter to once per 25.5 ms frame. In this section, we will investigate the effects of specifying the energy in the residual,and adjusting the quantizer accordingly, more than once per frame.

Utterances were processed with the adaptive noise shaping produced by a 1-zero filter. Versions were generated with each of the frames divided into 1, 5, and 10 sections and the energy specified for each section. This specification required an extra 2 bits per section.

By updating the energy parameter at a faster rate, the noise power will track the residual energy more closely. The noise spectral density will not vary within a frame as the analysis and synthesis filter coefficients are not changed. Each section within a frame will now have the same SNR. Listening tests show that the difference between utterances with 10 sections per frame and 1 section per frame is barely detectable. We conclude that for this system, time domain noise shaping on a short-time basis does not improve quality.

5. FREQUENCY DOMAIN NOISE SHAPING

Shaping of the noise power spectral density for each frame can be accomplished easily within the framework of the APC system. Although the system without any noise shaping, resulting in white noise error signal over the duration of each frame, will be optimal in the sense of minimal mean square error (MSE), it will not be optimal perceptually. Although experiments have shown that MSE is not a good indicator of quality, another metric that is a good indicator has not been found.

From previous work, it is known that the shaping of the noise spectrum to perceptually minimize the detectability of the noise should be a function of the input speech signal [6,7,8,9]. The estimate of the speech spectrum given by the linear prediction analysis, can be used to generate possible shaping filters for the encoding error. Only those spectral shapes for modifying the noise that can be derived from the all-pole linear prediction estimate of the speech short-time spectrum are investigated here.

5.1 All-Pole Noise Spectral Shaping

The first type of noise shaping implemented is specified by the all-pole equation,

$$B_1(z) = A^{-1}(\frac{z}{\alpha}) = \frac{1}{1+\sum_{k=1}^{p} a_k \alpha^k z^{-k}} \quad (3)$$

where A(z) is the optimal inverse filter for each frame from the linear prediction analysis. The poles of $B_1(z)$ have the same frequency as the poles of $A^{-1}(z)$ but have larger bandwidths. The bandwidth increase is given by:

$$\alpha = e^{-\pi f/Fs} \quad (4)$$

where f is the BW increase and Fs is the sampling frequency in hertz. Note that for $\alpha=0$, $B_1(z)$ is unity, i.e. no shaping. For $\alpha=1$, $B_1(z)$ is exactly the all-pole estimate of the speech spectrum. This APC noise shaping system is implemented as in Fig. 2.

The shape of the noise that results from a filter with poles at the same frequencies but moved closer to the origin in the z-plane as the poles of the linear prediction estimate of the speech signal is a spectrum similar to the speech spectrum but with increased formant bandwidths. Small increases in the bandwidths will not change the general overall spectral shape. Larger increases will tend also to flatten the spectrum.

Listening results indicate that increases of 200 Hz in the bandwidths give a rough quality to the speech. A noise shaping corresponding to an increase of the bandwidths of 800 Hz produces

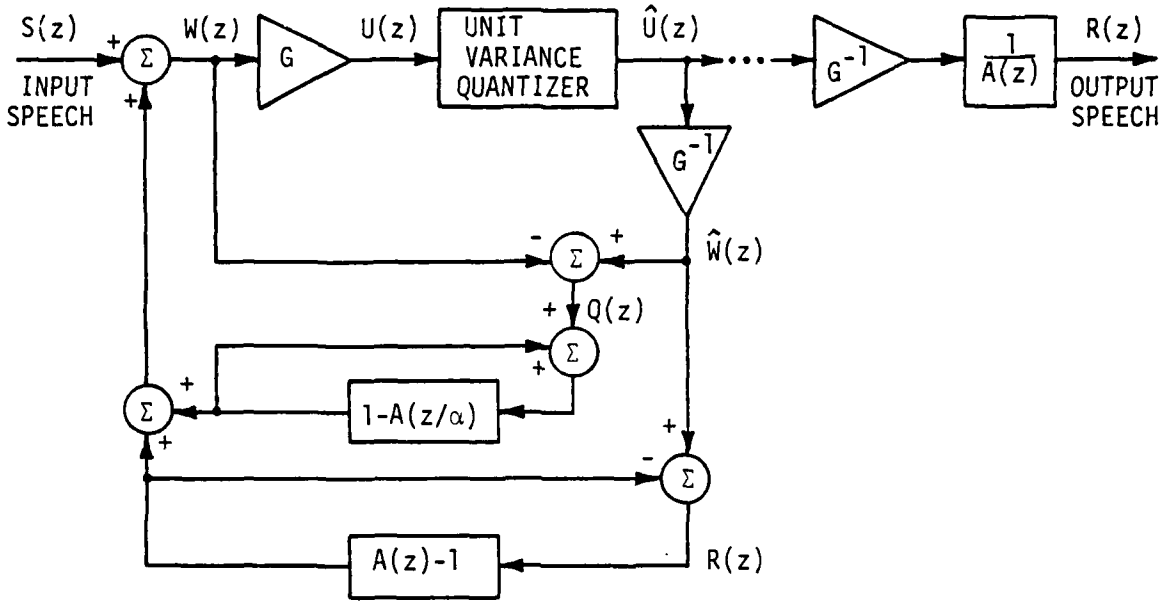


FIG. 2. APC-NS SYSTEM WITH $B_1 = A^{-1}(z/\alpha)$

a very slight roughness but has less audible noise than with the 1-zero shaping. Overall, the quality is judged to be better than the 1-zero noise shaping.

5.2 Fixed Pole, Variable Zero Noise Shaping

Another type of spectral noise shaping that was evaluated is generated by a filter of the form

$$B_2(z) = \frac{A(z/\beta)}{A(z)} = \frac{1 + \sum_{k=1}^{p} a_k \beta^k z^{-k}}{1 + \sum_{k=1}^{p} a_k z^{-k}} \qquad (5)$$

If $\beta=1$, this becomes the APC system without noise shaping. If $\beta=0$, $B_2(z) = A^{-1}(z)$, and the noise is shaped to the all-pole

estimated speech spectrum. The bandwidth increase produced for the zeros is given by Equation 4. As in the previous section, small increases in the bandwidth of a singularity will tend to smooth the spectral peak or valley but will leave the overall slope intact. Larger increases will flatten that overall slope. This implies that adding zeros at the same frequency as the poles but with a small bandwidth increase will remove the overall slope and leave only the sharp peaks in the spectrum. As the bandwidths are increased by moving the zeros further toward the origin in the z-plane, the spectrum will change in slope. When the zeros reach the origin (i.e.β=0 or infinite bandwidth increase), the spectrum will be the original all-pole speech spectrum. This effect can be seen in Fig. 3,4, and 5.

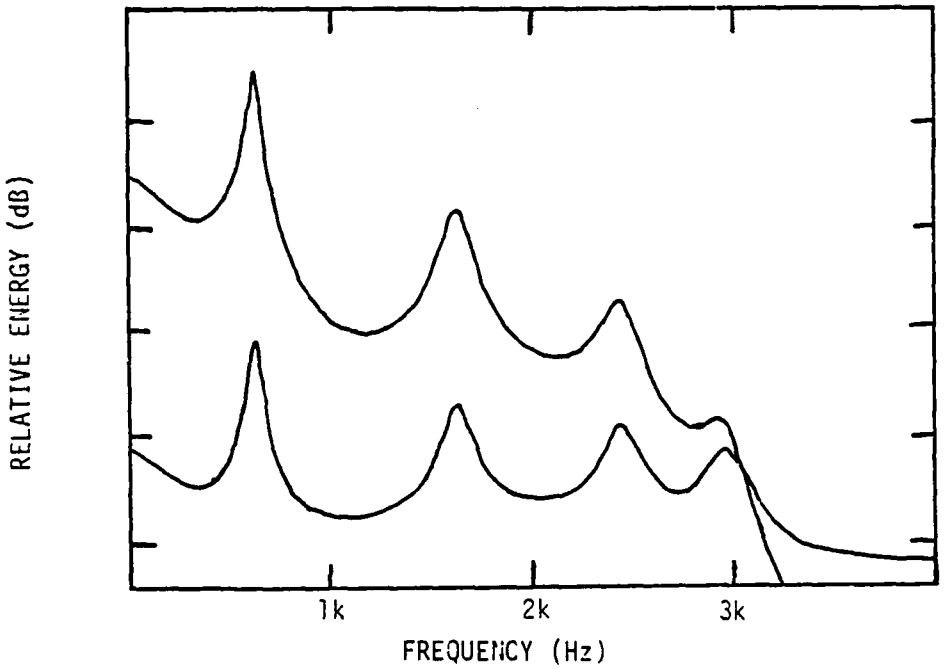


FIG. 3 ENCODING NOISE PRODUCED BY 200 HZ BW INCREASED BASED ON EQUATION 5

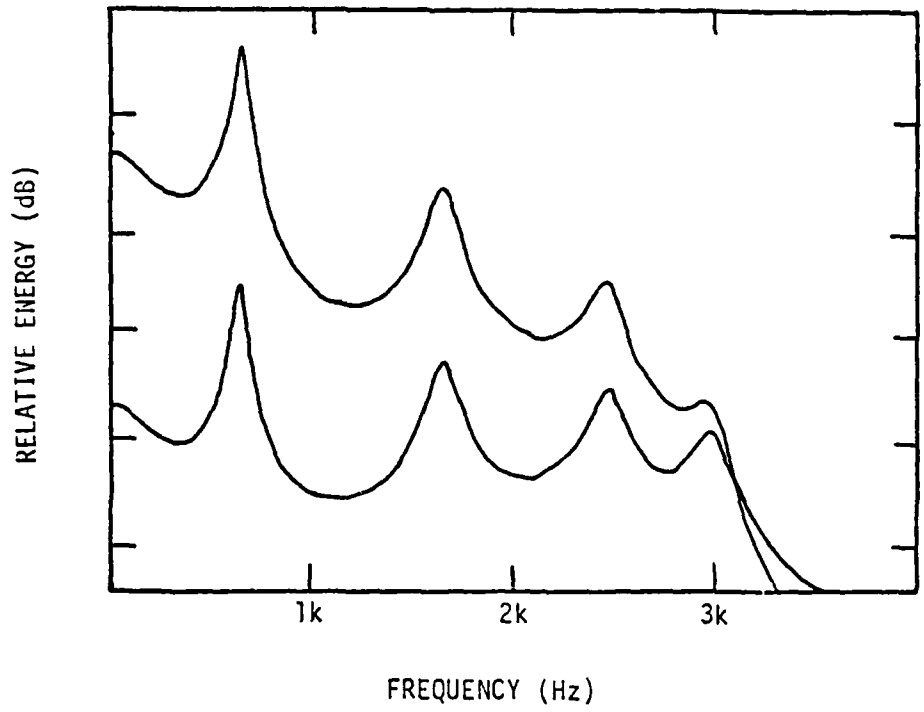


FIG. 4 ENCODING NOISE PRODUCED BY 400 HZ BW INCREASE BASED ON EQUATION 5

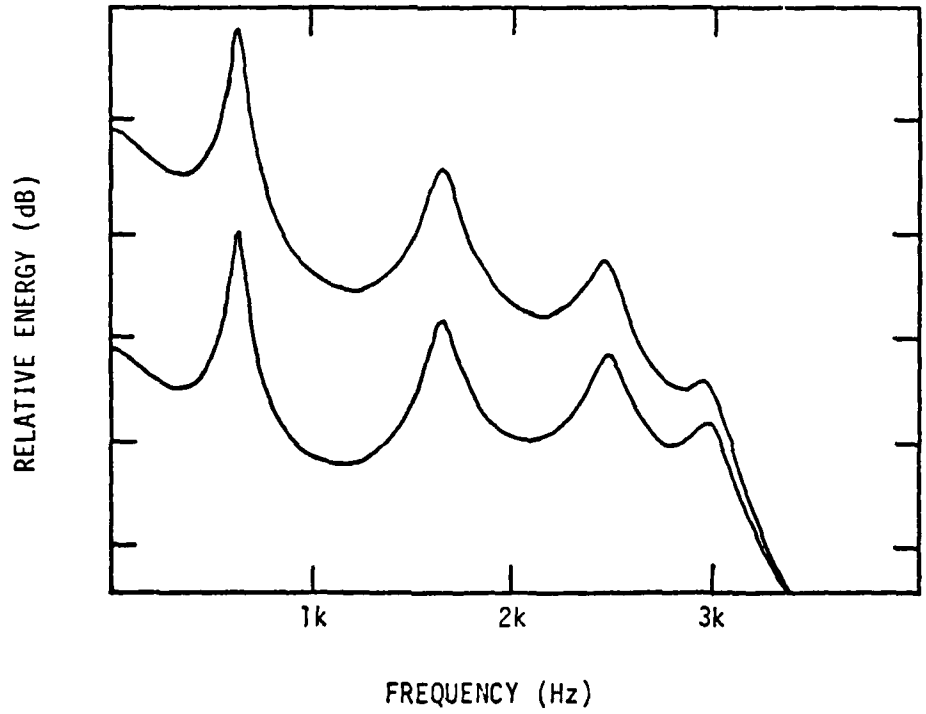


FIG. 5 ENCODING NOISE PRODUCED BY 800 HZ BW INCREASE BASED ON EQUATION 5

This scheme was implemented as in Figure 6. Ten utterances were processed with this noise shaping for a variety of values of bandwidth increases. For a bandwidth increase of 200 Hz, the speech sounded as if it had high frequency preemphasis and was noisy. For a bandwidth increase of 1600 Hz, this preemphasis effect is not present. The loudness of the noise decreased and the processed utterances were better than with the other noise shapings that were evaluated.

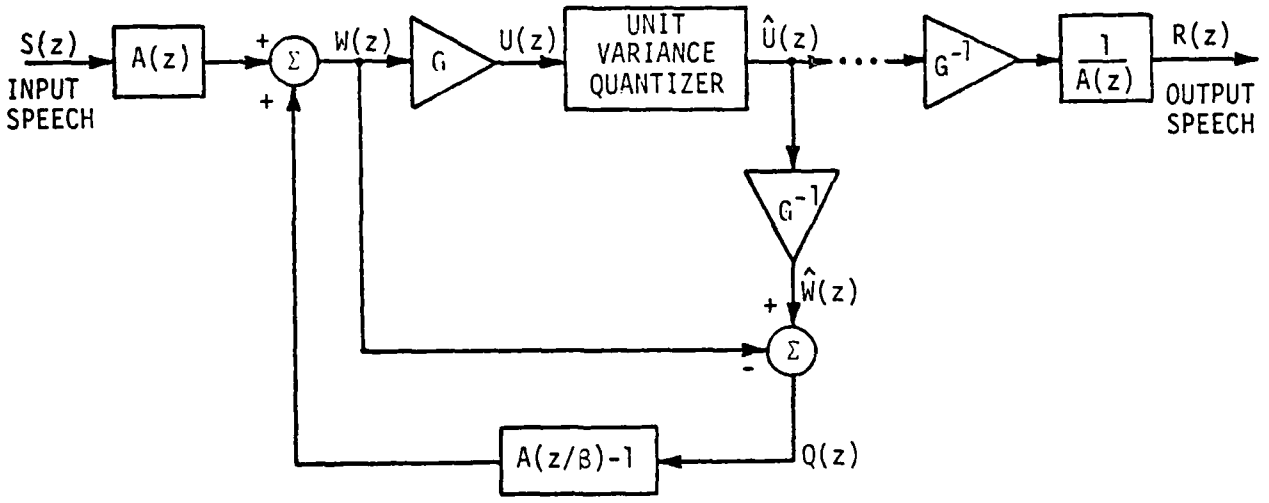


FIG. 6. APC-NS SYSTEM WITH $B_2 = A(Z/\beta)/A(Z)$

5.3 General Pole-Zero Noise Shaping

The generalization of the scheme in the previous section is to allow the poles to vary in bandwidth as well as the zeros. We have considered only shapings, however, where the zeros are

closer to the origin (larger bandwidth increase) than the poles. Shapings that do not fit this restriction have spectra that are approximations to the inverse of the speech spectrum, having formant valleys instead of peaks.

The system was implemented as in Fig. 7. The output of this system is:

$$R(z) = S(z) + \frac{A(z/\beta)}{A(z/\alpha)} Q(z) \qquad (6)$$

This particular implementation of the APC noise shaping system was chosen because it allows separate implementation of the zeros and the poles in the system. Other implementations require a cascade of several filter blocks in a single feedback path. For those implementations, there exist problems caused by the necessity of ordering the filters. This ordering is important because the updating of the filter coefficients at frame boundaries does not account for the memory inherent in the filters.

Fifteen different combinations of pole and zero bandwidth increases were evaluated. The increases for the poles varied from 0 to 1600 Hz and for the zeros, from 200 Hz to infinite (i.e. zeros at the origin).

Informal listening tests show that the preferred noise shaping was produced with no bandwidth increase for the poles and

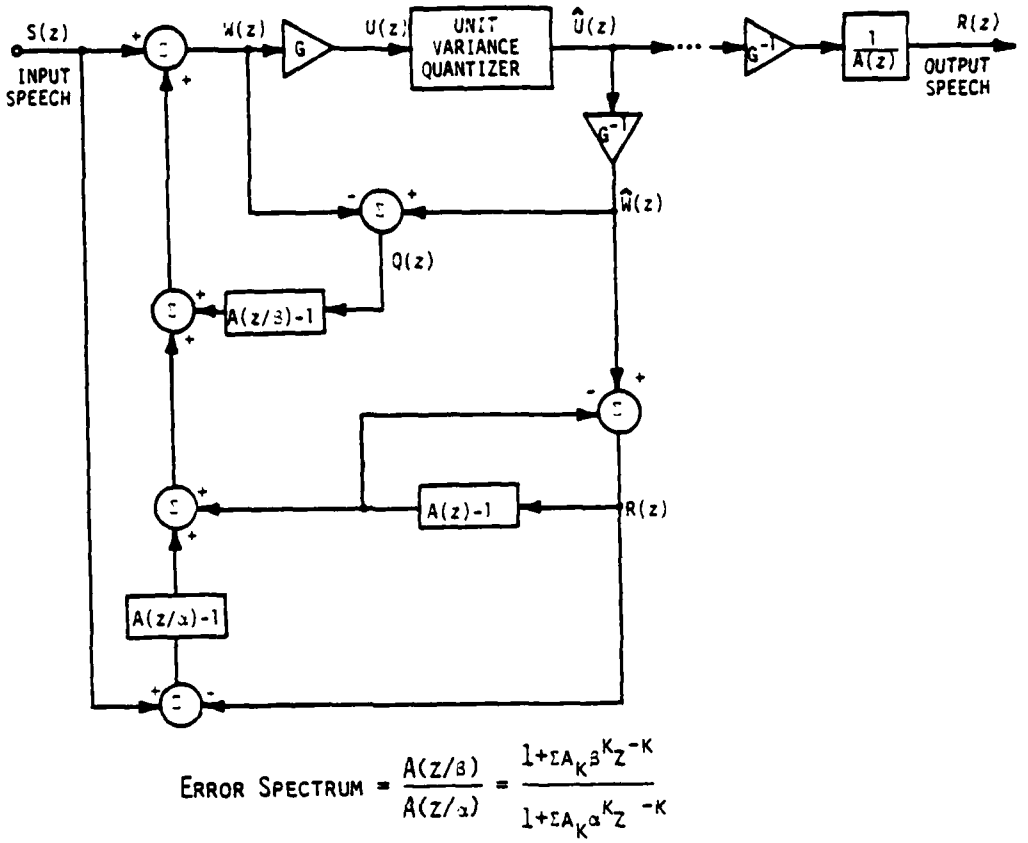


FIG. 7. APC-NS SYSTEM WITH $B_3 = A(Z/\beta)/A(Z/\alpha)$

an 1600 Hz increase for the zeros. This corresponds to the same shaping as was described in the previous section. Comparison of the implementations of Figures 2 and 7 for that noise shaping show no perceptible differences.

5.4 Instabilities of the APC Loop

The processing of utterances with the APC system both with and without noise shaping and using either fixed-length or

variable-length codes often produces "glitches" and "beeps", i.e., frames with negative signal to noise ratios. These severe degradations have been traced to the instability of the APC loop for many sets of coefficient values used in the filters in the feedback paths. Although the autocorrelation method used to find the prediction filter and to generate the feedback filters guarantees that the all-pole filter will be stable, it does not guarantee the stability of the APC loop itself.

The problem and some methods for its elimination are described here. The APC system is a feedback system employing negative feedback to produce a feedback error signal that is the input to the quantizer. If the blocks around the loop have a combined gain with magnitude greater than unity and phase of zero at some frequency, the loop will be unstable. In general, if the gain around the loop to a white noise signal is greater than unity, the system will be unstable. To evaluate the loop stability, let us look at the component blocks in the APC system without noise shaping as shown in Fig. 8. For ease of analysis, the noise feedback configuration (as was used in Fig. 6) is depicted.

The gain of the quantizer in terms the ratio of quantizer output noise power to quantizer input power is just the inverse of the SNR. A signal at the input to the quantizer block will

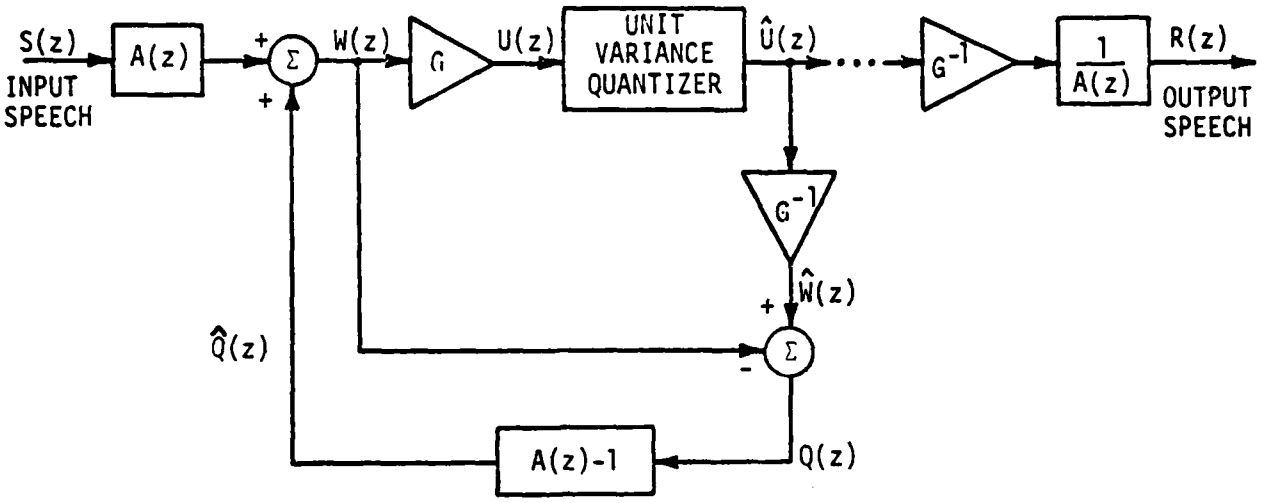


FIG. 8. APC SYSTEM WITHOUT NOISE SHAPING

produce a noise component with a gain dependent upon the quantization scheme. The variable length coding used here at an average of 2 bits per sample yields a signal to noise ratio for the quantizer of 10 to 12 dB, the exact value for a frame being a function of the sample value distribution in that frame. Thus, the gain for the quantizer block varies from -10 to -12 dB.

The feedback term to the summing block is

$$\hat{Q}(z) = (A(z)-1)\, Q(z) \quad .$$

If we assume that the quantization error is a white noise and independent of the input, the power gain of the feedback filter

for the quantization error signal is the sum of the squares of the predictor coefficients. This value has been found to be as large as 16 dB for this system for the utterances we have processed. When the product of the power gains around the APC loop, the product of the filter power gain for the quantization error and the quantizer gain defined above, is greater than unity (0 dB), the system may be unstable. Experimental results confirm this result.

Insight into the problem can be gained by an examination of its cause. The large power gain in the predictor filter is caused by a large spectral dynamic range of the speech signal. The high frequency portion of the speech is a spectral region of low energy and is a major contributor to the spectral dynamic range. In part, this is caused by the general spectral roll-off of speech and the non-ideal anti-aliasing filter used in the digitizing process.

There are several ways to avoid instability of the APC loop all by the general method of decreasing the loop gain. Increasing the quantizer input-to-error ratio would require a larger number of bits for the quantizer and will not be discussed. The power gain of the feedback filter can be lowered by either modifying that filter or by adding a limiter block or an adaptive gain at its output.

The method of adding a limiter or an adaptive gain into the loop is simplest in the system implementation of Fig. 6, requiring no additional modification. This addition into the loop will however, change the spectral shape of the noise.

Another method that is based on modification of the predictor filter coefficients is a high frequency correction scheme [7]. A term that corresponds to a high frequency signal with energy proportional to the speech residual is added to the autocorrelation vector, equivalent to adding to the high frequency region of the speech power spectrum. Experimental results show that the resultant power gain is lowered significantly. A typical utterance with a maximal power gain of 15.15 dB without high frequency correction had a maximum of 5.40 dB with the correction. Degradations due to instabilities were eliminated.

5.5 Conclusion

Informal listening tests have shown that spectral noise shaping significantly improves the quality of the APC encoded speech utterances. The preferred noise shaping was that given by a filter as specified by (5) and demonstrated in Fig. 5. The implementation may be either of the form of Figure 6 or 7 depending upon other requirements.

Instabilities in the APC loop can be a major source of degradation. Modification by high frequency correction has been found to eliminate these instabilities. This method of eliminating the instabilities may be used at each frame or just at those frames which are deemed unstable or nearly unstable.

6. COMPUTATIONAL COMPLEXITY OF ALGORITHM

The assessment of the computational complexity of the APC algorithm is simplified by noting that the vast majority of the computation occurs in just a few subroutines and operations. The emphasis of this section, therefore, is on those functional blocks in the algorithm.

The computational operations that were examined were addition, multiplication, division, and memory references. Because the implementation of this algorithm is via a Fortran program, the number of each of these operations that the computer will actually execute is highly dependent upon the Fortran compiler used. As this examination is for the purpose of assessing the feasibility of implementation of the algorithm on other computers, the degree of quantification here should suffice. Also, the examination will point out those subroutines and functional blocks whose optimization will affect the execution time of the algorithm.

Table 2 contains the approximate number of times each operation will be executed for the functional blocks given in Fig. 9. The information is given at the functional rate (e.g. once per point or once per frame) It is also given as normalized to the input sampling frequency of 8 kHz or operations to be executed every 125 usec.

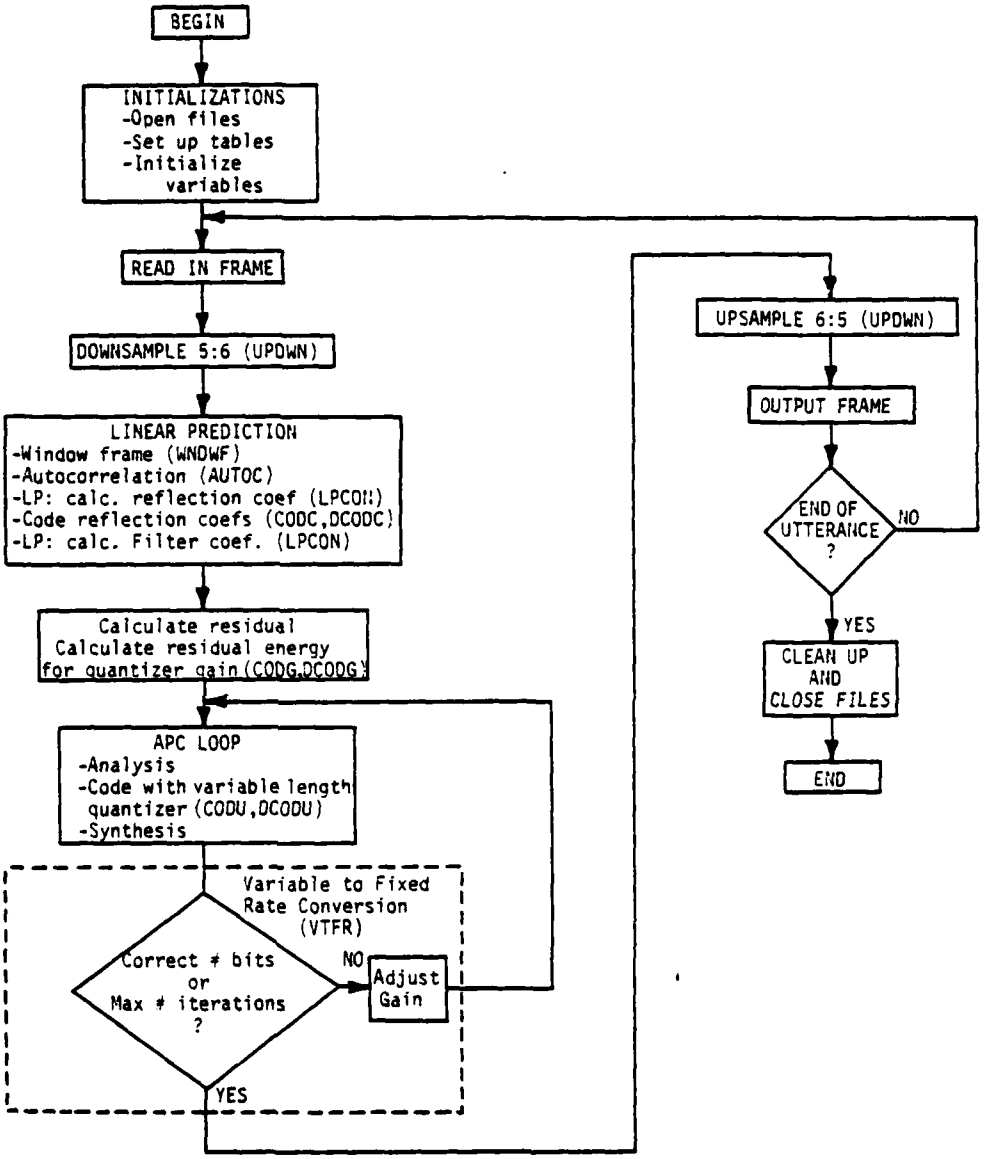


FIG. 9. FLOW CHART OF APC SYSTEM

TABLE 2. NUMBER OF OPERATIONS PER FUNCTIONAL BLOCK

FUNCTION	FREQ OF OPERATIONS	# OF OPERATIONS +	*	/	mem ref	# OPERATIONS/ INPUT POINT AT 8 KHZ +	*	/	mem ref
downsample	1/pt = 8k/s	295	85	0	672	295	85	0	672
window	1/pt = 6.67k/s	1	1	0	4	1	1	0	4
autocorrelate	1/pt = 6.67k/s	24	8	0	32	20	7	0	27
calc refl coef	1/25.5 ms frame	320	128	8	1408	2	1	0	7
code refl coef	1/25.5 ms frame	64	16	40	232	0	0	0	1
calc filt coef	1/25.5 ms frame	320	136	10	1408	2	1	0	7
calc residual	1/pt = 6.67k/s	28	10	0	68	24	9	0	57
code gain	1/25.5 ms frame	5	1	1	16	0	0	0	0
APC loop	5/pt= 33.3k/s	95	33	1	174	396	138	4	725
upsample	1/pt = 6.67k/s	350	101	0	800	292	84	0	667
					TOTALS:	1032	326	4	2167

Analysis of the table shows that approximately 60% of the computation is involved in the resampling operations, downsampling from 8 kHz to 6.67 kHz before the APC processing and upsampling from 6.67 kHz to 8 kHz after the processing. The APC loop itself accounts for the majority of the other computation. It is these two functions that should be examined if a more computationally efficient implementation of the algorithm is desired.

6.1 Possible Efficiencies for the Resampling Routine

The resampling subroutine, UPDWN, can be made to execute faster by several changes. The amount of computation is directly proportional to the length of the filter used in the routine. It may be possible to shorten the present filter, a symmetric FIR filter of order 250, without introducing any audible aliasing by using an IIR filter or a different FIR filter.

Of the nearly 600 additions per input point in the resampling, approximately 500 are used for indexing into arrays during the convolution of the signal with the filter. Auto-incrementing and auto-decrementing index registers could eliminate those additions. Use of several index registers would also decrease the number of memory references by a similar amount.

6.2 Possible Efficiencies for the APC Loop

The other functional block to be examined is the APC loop. As in the resampling operation, many of the additions are used in indexing. Auto-incrementing and auto-decrementing index registers would eliminate execution of those additions.

The computations shown in Table 2 are based on the implementation of the block diagram of Fig. 7. The noise shaping

found to sound best can be implemented by either that method or by the method shown in Fig. 6, the noise feedback configuration. Because there is only one feedback filter path, the noise feedback configuration is more efficient in terms of computation. The residual has already been computed as it was needed to compute the quantizer gain. It is necessary to have a separate synthesis block, but it is only performed once per frame, not each iteration of the variable to fixed rate conversion scheme. Assuming five iterations of the variable to fixed rate conversion scheme, the number of operations for the noise feedback configuration is approximately 138 additions, 45 multiplications, 4 divisions, and 304 memory references per input point at 8 kHz sampling rate. This represents a saving of about 60% for the APC loop.

The amount of computation for the APC loop is nearly proportional to the number of iterations needed for the variable to fixed rate conversion. The number of iterations was set at five as a compromise between computational load and increasing the number of bits used for coding rather than wasted as filler bits. It has been shown that the conversion scheme will converge better when the power gain of the predictor is lowered by the high frequency correction method. Fewer iterations may then be needed to attain the same degree of convergence as before. In any case, if the computational requirements are too high, they

may be reduced by allowing fewer iterations at a cost of fewer bits being used for quantization.

7. CONCLUSIONS AND PLANS FOR FURTHER WORK

From Section 6, the major conclusion to be drawn at the end of this first quarter is that the system is computationally quite expensive. Most of the computations are due to (i) the downsampling and upsampling operations between the 8 kHz and the 6.67 kHz sampling rates, and (ii) the iterative convergence algorithm to convert from variable to fixed rate. For the future, we shall direct our efforts toward developing computationally simpler schemes. To that end, we shall experiment with combinations of one or more of the following: (i) Processing the 8 kHz data without downsampling, (ii) Backward adaptation of the system parameters so that they do not need to be transmitted, and (iii) Bit allocation over frames of residual samples using fixed-length binary codes. From our past experience and from on-going experiments, we have evidence that the quality of the coded speech will suffer somewhat when using one of the above three proposed schemes. Therefore, it is imperative that we continue our work on improving speech quality. In particular, we shall implement and test the pitch loop and continue our efforts on time and frequency domain noise shaping. Our future experiments will be done mainly with the noise-corrupted speech data base.

8. REFERENCES

[1] M. Berouti and J. Makhoul, "Proposal for the Efficient Encoding and Decoding of Speech," Bolt Beranek and Newman Proposal No. P72-ISD-63, Solicitation No. MDA904-79-R-0826, May 1979.

[2] H. Gish and J.N. Pierce, "Asymptotically Efficient Quantizing," IEEE Trans on Information Theory, Vol. IT 14, Number 5, Sept. 1968.

[3] P. Noll and R. Zelinski, "Bounds on Quantizer Performance in the Low Bit-Rate Region," IEEE Trans on Communications, Vol.Com-26, No. 2, Feb. 1978.

[4] M. Berouti and J. Makhoul, "Efficient Encoding and Decoding of Speech: Final Report," Bolt Beranek and Newman Report No. 4063, Contract No. MDA904-76-C-0507, Feb. 1979.

[5] R. Hamming, "Introduction to Applied Numerical Analysis," McGraw-Hill Book Company, 1971.

[6] R. Zelinski and P. Noll, "Adaptive Transform Coding of Speech Signals," IEEE Trans. on Acoust. Speech and Signal Proc., Vol. ASSP-25, No. 4, Aug. 1977.

[7] B.S. Atal and M.R. Schroeder, "Predictive Coding of Speech Signals and Subjective Error Criteria," IEEE Trans. on Acoust. Speech and Sig. Proc., Vol. ASSP-27, No. 3, June 1979.

[8] J. Makhoul and M. Berouti, "Adaptive Noise Spectral Shaping and Entropy Coding in Predictive Coding of Speech," IEEE Trans. on Acoust. Speech and Signal Proc., Vol. ASSP-27, No. 1, Feb. 1979.

[9] M.A. Krasner, "Digital Encoding of Speech and Audio Signals Based on the Perceptual Requirements of the Auditory System," M.I.T. Lincoln Laboratory, Technical Report No. 535, Contract No. F19628-78-C-0002, June 1979.